Srtf:-

#include <stdio.h>

int main()

{

int arrival\_time[10], burst\_time[10], temp[10];

int i, smallest, count = 0, time, limit;

double wait\_time = 0, turnaround\_time = 0, end;

float average\_waiting\_time, average\_turnaround\_time;

printf("\nEnter the Total Number of Processes : \t");

scanf("%d", &limit);

printf("\nEnter Details of %d Processes\n", limit);

for (i = 0; i < limit; i++)

{

printf("\nEnter Arrival Time : \t");

scanf("%d", &arrival\_time[i]);

printf("Enter Burst Time : \t");

scanf("%d", &burst\_time[i]);

temp[i] = burst\_time[i];

}

burst\_time[9] = 9999;

for (time = 0; count != limit; time++)

{

smallest = 9;

for (i = 0; i < limit; i++)

{

if (arrival\_time[i] <= time && burst\_time[i] < burst\_time[smallest] && burst\_time[i] > 0)

{

smallest = i;

}

}

burst\_time[smallest]--;

if (burst\_time[smallest] == 0)

{

count++;

end = time + 1;

wait\_time = wait\_time + end - arrival\_time[smallest] - temp[smallest];

turnaround\_time = turnaround\_time + end - arrival\_time[smallest];

}

}

average\_waiting\_time = wait\_time / limit;

average\_turnaround\_time = turnaround\_time / limit;

printf("\n\nAverage Waiting Time : \t%.3lf\n", average\_waiting\_time);

printf("Average Turnaround Time : %.3lf\n", average\_turnaround\_time);

return 0;

}

Q2)- LRU

#include <stdio.h>

int findLRU(int time[], int n)

{

int i, minimum = time[0], pos = 0;

for (i = 1; i < n; ++i)

{

if (time[i] < minimum)

{

minimum = time[i];

pos = i;

}

}

return pos;

}

int main()

{

int no\_of\_frames, no\_of\_pages, frames[10], pages[30], counter = 0, time[10], flag1, flag2, i, j, pos, faults = 0;

printf("Enter number of frames: ");

scanf("%d", &no\_of\_frames);

printf("Enter size of the incoming stream : ");

scanf("%d", &no\_of\_pages);

printf("Enter the stream : ");

for (i = 0; i < no\_of\_pages; ++i)

{

scanf("%d", &pages[i]);

}

for (i = 0; i < no\_of\_frames; ++i)

{

frames[i] = -1;

}

for (i = 0; i < no\_of\_pages; ++i)

{

flag1 = flag2 = 0;

for (j = 0; j < no\_of\_frames; ++j)

{

if (frames[j] == pages[i])

{

counter++;

time[j] = counter;

flag1 = flag2 = 1;

break;

}

}

if (flag1 == 0)

{

for (j = 0; j < no\_of\_frames; ++j)

{

if (frames[j] == -1)

{

counter++;

faults++;

frames[j] = pages[i];

time[j] = counter;

flag2 = 1;

break;

}

}

}

if (flag2 == 0)

{

pos = findLRU(time, no\_of\_frames);

counter++;

faults++;

frames[pos] = pages[i];

time[pos] = counter;

}

printf("\n");

for (j = 0; j < no\_of\_frames; ++j)

{

printf("%d\t", frames[j]);

}

}

printf("\n\nTotal Page Faults = %d", faults);

return 0;

}